Speech classifier using SVM linear model/ Naïve Bayes/ Neural Network

Abstract

For this project we worked on three different classification approaches: LSTM, Naïve Bayes, and SVM. Related research indicates that LSTM should be the most effective one of these at sentiment analysis tasks, so we were interested in devising and performing an experiment which compares these three common machine learning approaches considering a multiclass (negative, neutral, positive) text classifying problem.To complete this task, we used two Amazon review datasets of 1 million entries each: one related to the topic of Movies for training and testing (an 80/20 split), and one about Sports for evaluation. The results from our project show that LSTM and SVM perform at similar levels of accuracy (76% vs 72% on testing, 70% vs 76% on evaluation), while Naïve Bayes is worse at the task than both of them (67% on testing, 61% on evaluation).

Introduction

The emergence and evolution of computers has in many ways changed the way the world works. From initially having the capabilities to compute, store and calculate, computers nowadays, with the help of the internet, offer the possibilities to do shopping, banking and communicate with people all over the world. Things that previously were done in person are now possible to do online, which have changed people's daily lives. Though, with these possibilities the needs for data have been increasing. Companies need data on their customers, what they buy and what they review etc (Kubina, Varmus and Kubinova, 2015). Additionally, there are many other fields using data to function including science, healthcare and geography (Favaretto et al., 2020).

With time, the ongoing collections of data have become many and the datasets have become big and often varied and complex, at many times too difficult for traditional databases to handle and process (Sagiroglu and Sinanc, 2013). The term used to describe this is ‘big data’ and big data analytics is the way of processing and attempting to extract value from this complex big data. This includes among others predictive analytics and user behavior analytics.

Predictive analytics in itself, has become more and more popular and important with the progress of social media. People are expressing feelings and opinions in many ways, through tweets, Facebook statuses, Reddit forums and also product reviews on web sites of product selling companies (Borras-Morell, 2015). This has given ground and potential for businesses and companies to discover what needs to be improved in their products and where their marketing strategies should lie, to save money and make profit. Thus, sentiment analysis, a subfield of natural language processing, is being used more and more to handle thousands of opinions and reviews out there.

The purpose of sentiment analysis is to identify sentiment and how it is expressed in texts. In addition it is about understanding what the expressions indicate in terms of positive, neutral or negative opinions (Nasukawa and Yi, 2003). This is done with many different approaches, among other machine learning techniques such as Naïve Bayes and Support Vector Machine (SVM) as well as deep learning techniques such as Long Short-term Memory neural networks. In this work we compare the performance of these three methods on training, testing datasets and examine their robustness by using an evaluation dataset totally distinct from training data. In terms of performance metrics we use: Accuracy, AUC, F1-Score, Precision and Recall.

Related research

A study from 2020 used sentiment analysis on Amazon product reviews. Differences in performances between a Support Vector Machine classifier and a Naïves bayes classifier were observed and compared. In the study's data processing, neutral reviews were discarded from the collection of reviews, leaving the positive and negative to remain, and thus it regarded binary classification. The Support Vector Machine was shown to be the better performing out of the two classification techniques used (Dey et al., 2020).

Another study conducted in 2019 investigated whether the three machine learning sentiment analysis techniques Linear Support Vector Machine (L-SVM), Long Short-Term Memory (LSTM) and Multinomial Naïve Bayes (MNB) were feasible to apply in the context of Amazon reviews. Additionally, they compared the performances of the techniques to identify which technique performed the best and was the most feasible. The sentiment techniques had the task of binary classification, either to determine if a review was positive or negative. The study concluded that the LSTM technique was the most well performing and compatible technique to binary sentiment analysis on different Amazon reviews (Coyne, Smit and Güner, 2019).

Wazery, Mohammed and Houssein (2018) used three Twitter datasets of IMDB, Amazon and Airline in their study to compare machine learning approaches and deep learning approaches. The machine learning methods included Naïve Bayes, Support Vector Machine, decision tree and K-nearest Neighbour while the deep learning approach solely included the Long short-term memory neural network. The results showed that the performance of classifying the tweets into either positive or negative was done the best by the LSTM neural network in terms of accuracy on balanced datasets.

Barry (2017) also compared Support Vector Machine, Naïve Bayes and Long Short-term Memory neural network on balanced datasets and showed, again in line with the other mentioned studies including these three classification methods, that the LSTM performed the best in binary sentiment classification.

To summarize the previous research, it is evident that the LSTM neural network performed the best in binary sentiment classification of balanced datasets. Yet, what all of these studies have in common is the exclusion of neutral reviews since the neutral reviews have intentionally been discarded.

Research question

The related research showed that LSTM neural networks performed better than Naïve Bayes and SVM classifiers in sentiment analysis. It was done in a binary sentiment setting and further room for investigation of performance between the three methods could be done by extending the datasets to contain positive, negative and neutral reviews.

Therefore, our project aims to explore the differences between Naïve Bayes, Support Vector Machine, and neural network (LSTM) models in terms of performance on a multiclass sentiment classification task.

This is done with the following research question:

Which classification approach out of Naïve Bayes, Support Vector Machine and LSTM neural network performs the best at evaluating datasets containing positive, negative and neutral opinions? Is the performance dependent on the topic of training review texts?

Hypothesis
Based on the previous research that we discussed earlier, our hypothesis going into this project was the following:

H_0. *All three of the models will perform similarly at the sentiment classification task.*
H_1. *A Support Vector Machine will perform better at the sentiment classification task than both LSTM and NB.*

To test our hypothesis, we constructed three different models in Python, utilizing various libraries such as sklearn, keras, pyTorch, and others.

Method

Data Preparation & Preprocessing
We used sections of two Amazon datasets in this project, the first one made up of movie reviews[1] and the second one of reviews for sports- and outdoor activity-related items[2]. Both original datasets were very unbalanced in favour of positive reviews, unlabelled, and too large for the processing power we had at our disposal. Due to this situation, the first steps we took were as follows:

- Assign classes to the datasets' entries such that 1 or 2 star reviews are labelled as 0 (negative), 3 star reviews are labelled as 1 (neutral), and 4 or 5 star reviews are labelled as 2 (positive).
- Extract even amounts of entries from each of the classes, such that the total from a single dataset is equal to or just exceeds 1 million. This means that we ended up with two balanced datasets of 1 million entries each: one to use for training and testing (Movies) and one for evaluation (Sports).

Afterwards, we preprocessed the newly obtained datasets to get them ready for training our three models. First, we dropped any empty rows, removed emoticons, non-alphanumeric symbols, HTML tags, multiple spaces, and punctuation; we removed URLs, usernames, and numbers.

All remaining text was converted to lowercase. Stopwords (those specified in the nltk English stopwords list - including the important word "not") were initially removed,

[1] Dataset used for training (80%) and testing (20%):
http://deepyeti.ucsd.edu/jianmo/amazon/categoryFilesSmall/Movies_and_TV_5.json.gz
[2] Dataset used for evaluation:
http://deepyeti.ucsd.edu/jianmo/amazon/categoryFilesSmall/Sports_and_Outdoors_5.json.gz

but later we found out that accuracy is better if we replace contractions such as "couldn't" with their full forms instead. The text was tokenized using NLTK word_tokenizer. Finally, all words shorter than two characters were dropped as well, as they are usually pronouns and very short word stems unlikely to carry any relevant meaning, especially after tokenization.

Naïve Bayes Classifier
The Naïve Bayes Classifier is a probabilistic classifier built on Bayes' theorem. It uses strong Naïve assumptions that each feature of a target class is independent and has equal contribution to the probability of a chosen sample to belong to a particular target class (ScienceDirect Topics, 2021). Essentially meaning that the classifier assumes that a feature given a class is not connected to any other feature and that all features together contribute to the probability of something appearing or existing. This later decides what particular class the sample belongs to.

Linear SVM
SVM stands for Support Vector Machine and is a kind of linear model used for various classification tasks. It creates a hyperplane or line, which it uses to separate the input data into classes, if such a separation is possible. A hyperplane is defined as a flat subset of an n-dimensional space, which has one less dimension than that space and divides it into two sectors which are not connected.

The algorithm finds the best line (or hyperplane) by choosing the titular support vectors: the points from each class closest to the line. The distances between these points and a potential split line are called a margin; an SVM chooses the best fit by maximising the margin. For non linearly separable data, a dimension can be added to allow for the needed computation. Linear separators found in this way can be projected back to the original dimensionality.

Training and testing:
The model was trained on 80% of a dataset of movie and TV reviews with 1 million entries. The remaining 20% were used for the testing. The first step after splitting was to vectorize with a TFIDF Vectorizer from the sklearn python library. Next, the linear SVC model was instantiated. The model was fitted on the training data and scored on the test data, achieving around 72% accuracy.

Hyperparameter adjustments:
Hyperparameters for this model were manually tested to achieve a better accuracy rate. The most influential parameter was C in the LinearSVC function, however in the final iteration of the model the default value for it produced the best results. TFIDF Vectorizer parameters did not have as large an effect on the accuracy.

Evaluation:
Evaluation is done in the same way as testing, by vectorizing and then scoring the data, this time from our Sports and Outdoors dataset of 1 million entries (this set was used specifically for evaluation). The accuracy for this set was 76%. The classification reports for both the testing and the evaluation can be found in the Results section of this paper.

Long short-term memory (LSTM) architecture
A Long Short-Term Memory (Hochreiter et al., 1997) network is a special kind of recurrent neural network, capable of learning long-term dependencies. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn. This is a behavior required in such a complex problem as text classification and other NLP problems.

The key to LSTMs is the cell state, which runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged (provides the ability to remember long-term dependencies). An LSTM cell has four gates (forget-, input-, cell- and output gate) to control the information flow and maintain the cell state in the model.

Before applying the LSTM network, we utilized an Embedding layer which converts all the different 200.000 words (appeared in our training dataset) into 300-sized representation vectors containing information about the meaning of the given word. In order to reduce training time and memory usage, we utilized the pre-trained GLoVe (Pennington et al., 2014) embedding weights which proved to be very efficient as a feature extraction method. In order to regularize our model, we used a Dropout layer directly connected to the output layer which produces the probabilities of the three different classes after applying a Softmax nonlinearity.

During training, we used the popular Adam optimizer and Cross Entropy Loss function. The different values of hyper-parameters significantly influenced the model's performance. For example, Figure 1. and Figure 2. show the training and validation loss functions with optimal and suboptimal parameters (which resulted in spectacular overfitting). After manually optimizing the hyper-parameters of the neural network, the following values proved to be the most efficient: learning_rate=0.001, batch_size = 512, lstm_hidden_size = 128, nr_epochs=10, dropout_probability=0.2.

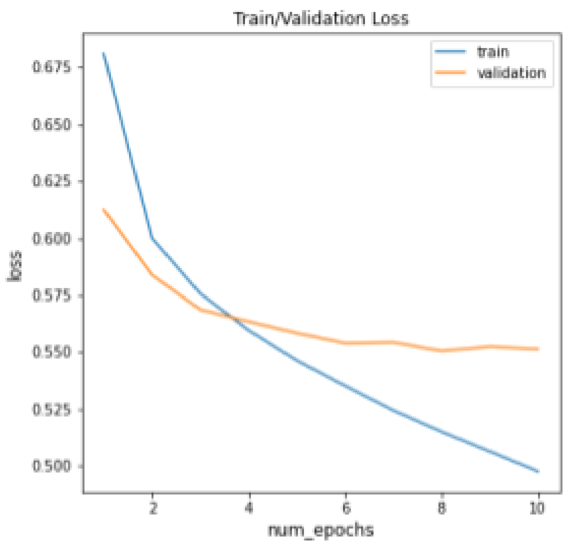


Figure 1. Loss values with optimal parameters

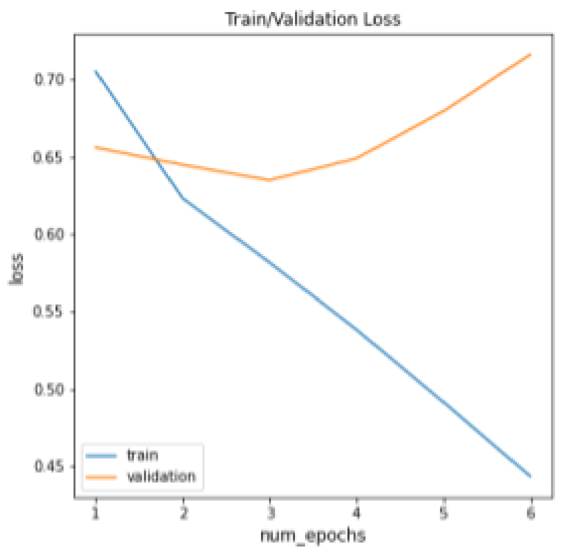


Figure 2. Loss values with suboptimal parameters

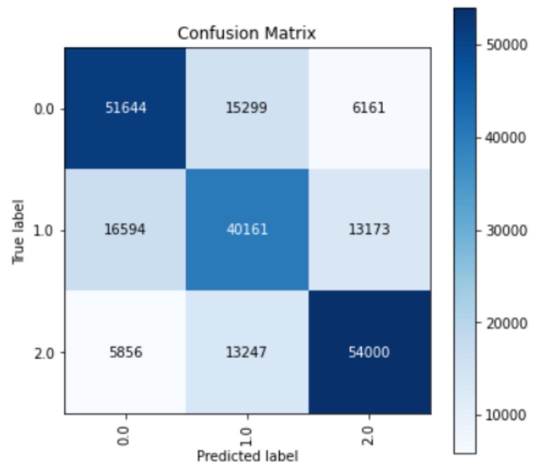
Results

Naïve Bayes

The Naïve Bayes classifier achieved a test accuracy of around 67% and an evaluation accuracy of 61%. The full classification reports are shown below, along with confusion matrices for both the testing and the evaluation phases.

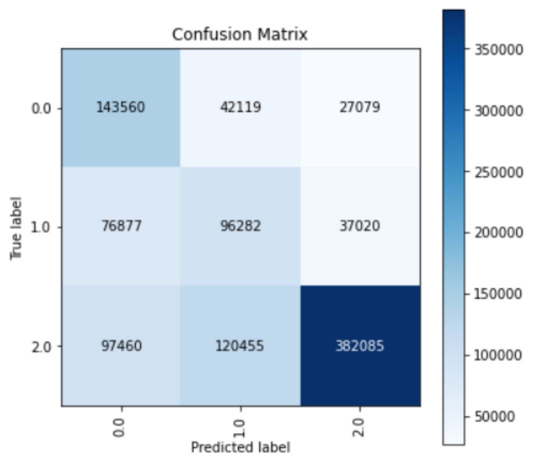
Testing

```
Accuracy: 0.6746015221967752
F1: [0.70169432 0.5793775  0.73751852]
Precision: [0.69700651 0.58452559 0.73635694]
Recall: [0.70644561 0.5743193  0.73868377]
AUC: 0.8462328645291564
```



Evaluation

```
Accuracy: 0.6079817232146261
F1: [0.54106717 0.41055358 0.73043556]
Precision: [0.45159281 0.37195197 0.85633954]
Recall: [0.67475724 0.45809524 0.63680833]
AUC: 0.7949740100845789
```



Linear SVM

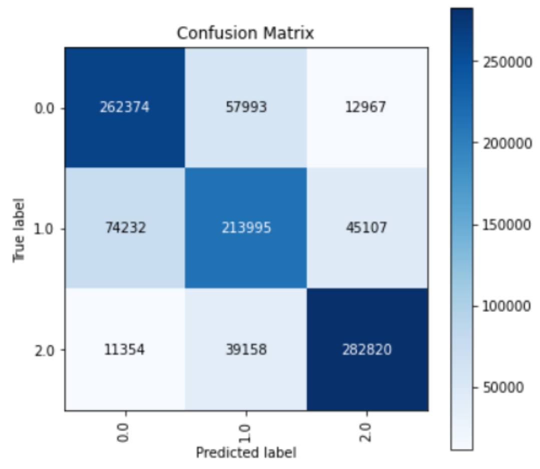
For the linear SVM model, the initial accuracy upon testing was 72% and improved slightly on the evaluation set to 76%. This was probably due to the overall relatively low accuracy of the training model: it is likely the accuracy increased by a small amount by sheer chance. The classification reports for both the testing and the evaluation are included below, and so is the confusion matrix.

Testing

```
Accuracy score: 0.7236
Classification report
              precision    recall  f1-score   support

         0.0       0.72      0.75      0.74     66535
         1.0       0.64      0.59      0.61     66449
         2.0       0.80      0.83      0.82     67016

    accuracy                           0.72    200000
   macro avg       0.72      0.72      0.72    200000
weighted avg       0.72      0.72      0.72    200000

--- 111.5530035495758 seconds ---
```

Evaluation

```
Accuracy score: 0.759189
Classification report
              precision    recall  f1-score   support

         0.0       0.75      0.79      0.77    333334
         1.0       0.69      0.64      0.66    333334
         2.0       0.83      0.85      0.84    333332

    accuracy                           0.76   1000000
   macro avg       0.76      0.76      0.76   1000000
weighted avg       0.76      0.76      0.76   1000000
```



Long short-term memory (LSTM)
Using the LSTM architecture, we could achieve an average of 76% test and 70% evaluation accuracy. In terms of AUC score, these values are 0.91 and 0.86 respectively. Table 1 contains the F1-score, Precision, Recall, AUC values for the three classes on both the test and evaluation dataset.
Figure 3 and Figure 4 show the confusion matrix generated on the test and evaluation dataset respectively.

	Test				***Evaluation***			
	Neg.	***Neutral***	***Pos.***	***Avg.***	***Neg.***	***Neutral***	***Pos.***	***Avg.***
Accuracy	0.76				*0.70*			
F1-Score	*0.79*	*0.66*	*0.82*	*0.76*	*0.66*	*0.50*	*0.81*	*0.66*
Precision	*0.79*	*0.67*	*0.83*	*0.763*	*0.63*	*0.42*	*0.91*	*0.65*
Recall	*0.80*	*0.66*	*0.83*	*0.763*	*0.69*	*0.61*	*0.73*	*0.68*
AUC	*0.93*	*0.85*	*0.95*	*0.91*	*0.9*	*0.79*	*0.91*	*0.86*

Table 1: Performance metrics of LSTM on test and evaluation datasets

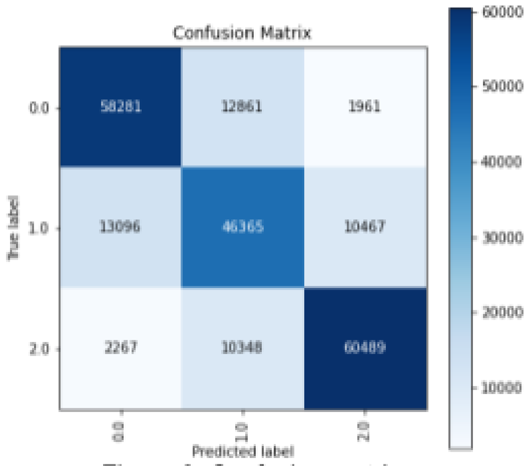


Figure 3: Confusion matrix on test dataset

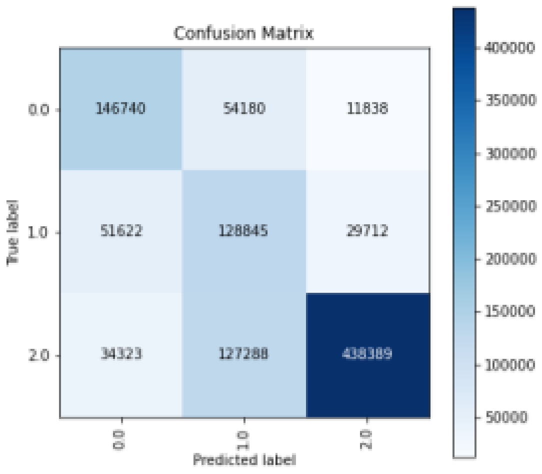


Figure 4: Confusion matrix on evaluation dataset

Discussion

Summary

In this project we aimed to test whether LSTMs perform better at sentiment classification tasks than SVMs and Naïve Bayes classifiers. After training, testing and evaluating all three models on the same two datasets, our results show that LSTM and SVM perform similarly while Naïve Bayes performs worse.

Therefore, the null hypothesis H_0 is rejected, as according to our experiment LSTM and SVM models performed the best, LSTM having a higher testing accuracy of 76%. SVM's accuracy was lower on testing (70%), while it was a little higher on evaluation, but given that these accuracies are not particularly high overall, the evaluation difference might be attributable to chance.

Challenges

At the start of the project, the Amazon datasets we found were unlabelled and very unbalanced, causing us to have to extract a balanced amount of entries from each class after completing the labelling. We chose to keep a total of 1 000 000 entries from the Movies & TV dataset and the same amount from the Sports and Outdoors dataset. Still, the remaining entries contained a large amount of different words (more than 500 000), so memory issues were common when applying CountVectorizer, a necessary part of implementing Naïve Bayes.

Neutral reviews proved to be the most difficult to get an accurate prediction on, accounting for most of the errors in the confusion matrices. For example, the SVM model had an accuracy of only 64% on neutral review predictions. This happens because of the lack of typical words with "neutral" connotations in comparison to "positive" and "negative" ones. Nevertheless, with consideration for the difficulty of the problem, all models achieved an acceptable accuracy rate overall.

Technical difficulties we encountered included high memory usage and a large number of hyperparameters in the case of LSTM. The sheer number of parameters meant a longer time was required in order to fine-tune this model. When it came to the SVM, the training time when using the SVC class from sklearn turned out to be extremely long and exponentially grew with the number of data points. However, replacing this class with its linear equivalent resolved the issue.

Algorithms
We used the three algorithms that seemed most promising to us based on the literature, but given more time, it would also have been interesting to explore other options such as a decision tree based algorithm, for example Random Forest or XGBoost.

Advantages and disadvantages of each algorithm:

- LSTM
  - Advantage: can handle long term dependencies, which proved to be useful in our case
  - Disadvantage: too many hyperparameters, hard to tune
- SVM
  - Advantage: simple and relatively fast (if using LinearSVC)
  - Disadvantage: has trouble representing more complex decision spaces
- Naïve Bayes
  - Advantage: simple implementation and works well on big datasets
  - Disadvantage: independence assumptions might not always coincide with reality and also it states that all features are equal

Contributions
In this project, we realized a data processing method which can be effectively applied in sentiment analysis problems and seems to improve overall accuracy in comparison with our initial data processing approach. We also highlighted the advantages and disadvantages of each of the three methods of multiclass sentiment analysis that we used. Finally, we examined how using a totally distinct evaluation dataset affects the performance and robustness of the three models.

Future research
It would be interesting to further explore the research question by improving and expanding our approach in the following ways:

- Continue tuning the hyper-parameters of each method (for example with an automatized hyper-parameter optimization tool, e.g. Optuna) to achieve even better results
- Use other datasets for evaluation and compare the results across topics and formats of text input
- Add more kinds of models to expand the comparison
- Train more than one of each model on different datasets and observe the differences in performance

Project assessment
Our team functioned very well together, achieving an even division of labour and providing each other with assistance whenever it was needed. Each team member implemented one of the three models and contributed to the writing of this report.

Conclusion

Our project found that LSTM and SVM are the better performing models. This is mostly in line with the literature on the three types of models, which indicates that LSTM is the best performing one. While the improvements listed above could benefit the project and even change the final results, we believe that our work sufficiently highlights the differences between the three models, their advantages and disadvantages, and their ranking when it comes to performance on sentiment classification tasks. From this project we conclude that there is some evidence that LSTM performs better than Naïve Bayes and similarly to SVM at these tasks, and that further exploration is required to achieve a deeper understanding of the differences between these models as well as expand the comparison to other approaches and dataset types.

References

Barry, J., 2017. Sentiment Analysis of Online Reviews Using Bag-of-Words and LSTM Approaches. In: *AICS*.

Borras-Morell, J.E., 2015. Data mining for pulsing the emotion on the web. *Methods in Molecular Biology (Clifton, N.J.)*, 1246, pp.123–130. https://doi.org/10.1007/978-1-4939-1985-7_8.

Coyne, E., Smit, J. and Güner, L., 2019. *Sentiment analysis for Amazon.com reviews*. https://doi.org/10.13140/RG.2.2.13939.37920.

Dey, S., Wasif, S., Tonmoy, D., Sultana, S., Sarkar, J. and Dey, M., 2020. *A Comparative Study of Support Vector Machine and Naïve Bayes Classifier for Sentiment Analysis on Amazon Product Reviews*. p.220. https://doi.org/10.1109/IC3A48958.2020.233300.

Favaretto, M., De Clercq, E., Schneble, C.O. and Elger, B.S., 2020. What is your definition of Big Data? Researchers' understanding of the phenomenon of the decade. *PLoS ONE*, 15(2), p.e0228987. https://doi.org/10.1371/journal.pone.0228987.

Kubina, M., Varmus, M. and Kubinova, I., 2015. Use of Big Data for Competitive Advantage of Company. *Procedia Economics and Finance*, 26, pp.561–565. https://doi.org/10.1016/S2212-5671(15)00955-7.

Nasukawa, T. and Yi, J., 2003. *Sentiment analysis: Capturing favorability using natural language processing*. p.77. https://doi.org/10.1145/945645.945658.

Pupale, Rushikesh. "Support Vector Machines(Svm) - an Overview." *Medium*, Towards Data Science, 11 Feb. 2019, https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989.

Sagiroglu, S. and Sinanc, D., 2013. Big data: A review. In: *2013 International Conference on Collaboration Technologies and Systems (CTS)*. 2013 International

Conference on Collaboration Technologies and Systems (CTS). pp.42–47. https://doi.org/10.1109/CTS.2013.6567202.

ScienceDirect 2021. *Naive Bayes Classifier - an overview | ScienceDirect Topics*. [online] Available at: <https://www.sciencedirect.com/topics/engineering/naive-bayes-classifier> [Accessed 12 Oct. 2021].

Wazery, Y.M., Mohammed, H.S. and Houssein, E.H., 2018. Twitter Sentiment Analysis using Deep Neural Network. In: *2018 14th International Computer Engineering Conference (ICENCO)*. 2018 14th International Computer Engineering Conference (ICENCO). pp.177–182. https://doi.org/10.1109/ICENCO.2018.8636119.

Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.